

# **CS 595: Assignment #6**

Due on Thursday, October 30, 2014

*Dr Nelson 4:20PM*

**Victor Nwala**

## Contents

<b>Problem 1</b>	<b>3</b>
<b>Problem 2</b>	<b>8</b>

## Problem 1

1. We know the result of the Karate Club (Zachary, 1977) split. Prove or disprove that the result of split could have been predicted by the weighted graph of social interactions. How well does the mathematical model represent reality?

Generously document your answer with all supporting equations, code, graphs, arguments, etc.

To answer this question, I firstly downloaded the Zachary's Karate Club GraphML file from the link:

<http://nexus.igraph.org/api/dataset?id=1&format=GraphML> and unzipped the GraphML file.

I also found a useful resource at this link:

<http://lists.nongnu.org/archive/html/igraph-help/2008-11/msg00047.html> created by Tamas Nepusz posted on Mon, 17 Nov 2008.

I used the GirvanNewman algorithm to split my graph, the algorithm does the following:

- 1)The betweenness of all existing edges in the network is calculated first.
- 2)The edge with the highest betweenness is removed.
- 3)The betweenness of all edges affected by the removal is recalculated.
- 4)Steps 2 and 3 are repeated until you get the required number of clusters.

Listing 1 shows a python script using GirvanNewman algorithm .

Listing 1: Python Script to Answer problem 1 and 2

```

from igraph import *
karate = load("karate.GraphML")
karate.vs["label"] = karate.vs["name"]
karate.vs["Faction"] = ["1", "1", "1", "1", "1", "1", "1", "1", "1", "2", "2", "1", "1", "1", "1",
    "2", "2", "1", "1",
5  "2", "1", "2", "1", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2"]
color_dict = {"1": "blue", "2": "pink"}
karate.vs["color"] = [color_dict[Faction] for Faction in karate.vs["Faction"]]
layout = karate.layout("kamada_kawai")
plot(karate, "original.pdf", layout = layout, bbox = (600, 600), margin = 40)
10
while len(karate.clusters()) <= 6:

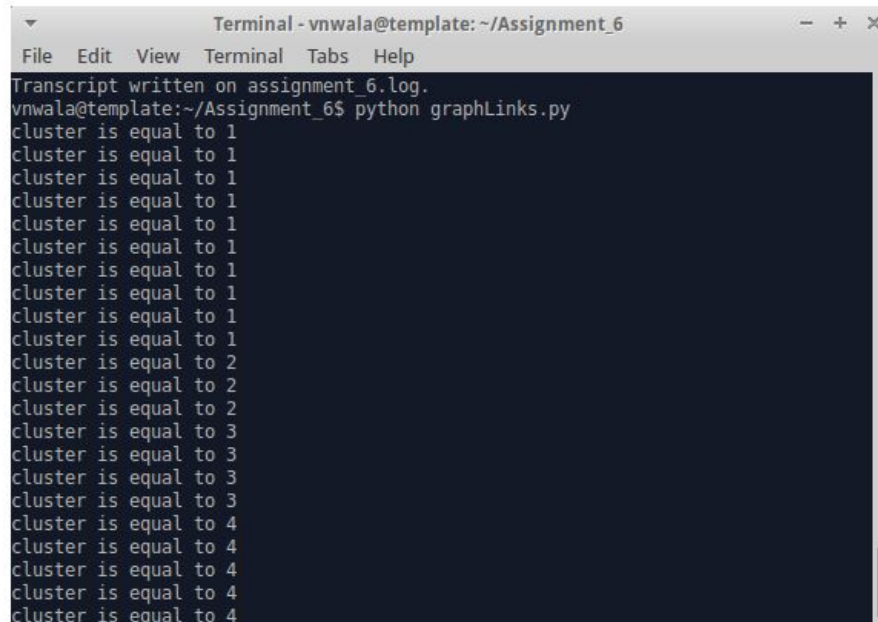
    ebs = karate.edge_betweenness()
    # determines the edge betweenness
15    max_idx = max(xrange(len(ebs)), key = ebs.__getitem__)
    # find the index of the edge with the maximum betweenness
    karate.delete_edges(max_idx)
    # remove that edge
    print "cluster is equal to" + ' ' + str(len(karate.clusters()))
20    # prints the number of clusters after an edge is removed
    if len(karate.clusters()) == 2:
        layout = karate.layout("kamada_kawai")
        plot(karate, "final_2.pdf", layout = layout, bbox = (600, 600), margin = 40)
        #saves plot when 2 clusters are reached
25    if len(karate.clusters()) == 3:
        layout = karate.layout("kamada_kawai")
        plot(karate, "final_3.pdf", layout = layout, bbox = (600, 600), margin = 40)
        # saves plot when 3 clusters are reached
    if len(karate.clusters()) == 4:
30    layout = karate.layout("kamada_kawai")
        plot(karate, "final_4.pdf", layout = layout, bbox = (600, 600), margin = 40)
        # saves plot when 4 clusters are reached

```

```
35 if len(karate.clusters()) == 5:
    layout = karate.layout("kamada_kawai")
    plot(karate, "final_5.pdf", layout = layout, bbox = (600, 600), margin = 40)
    # saves plot when 5 clusters are reached
```

This code plots the original graph with a color code of the factions predicted by the graph before split and the iterates to split the graph from initial cluster of 1 to the desired number of clusters specified by the code.

Figure 1: Code to plot graph and reduce graph to desired clusters



```
Terminal - vnwala@template: ~/Assignment_6
File Edit View Terminal Tabs Help
Transcript written on assignment_6.log.
vnwala@template:~/Assignment_6$ python graphLinks.py
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 1
cluster is equal to 2
cluster is equal to 2
cluster is equal to 2
cluster is equal to 3
cluster is equal to 3
cluster is equal to 3
cluster is equal to 3
cluster is equal to 4
cluster is equal to 4
cluster is equal to 4
cluster is equal to 4
```

graphLinks.py at work

Figure 2: Original plot before split

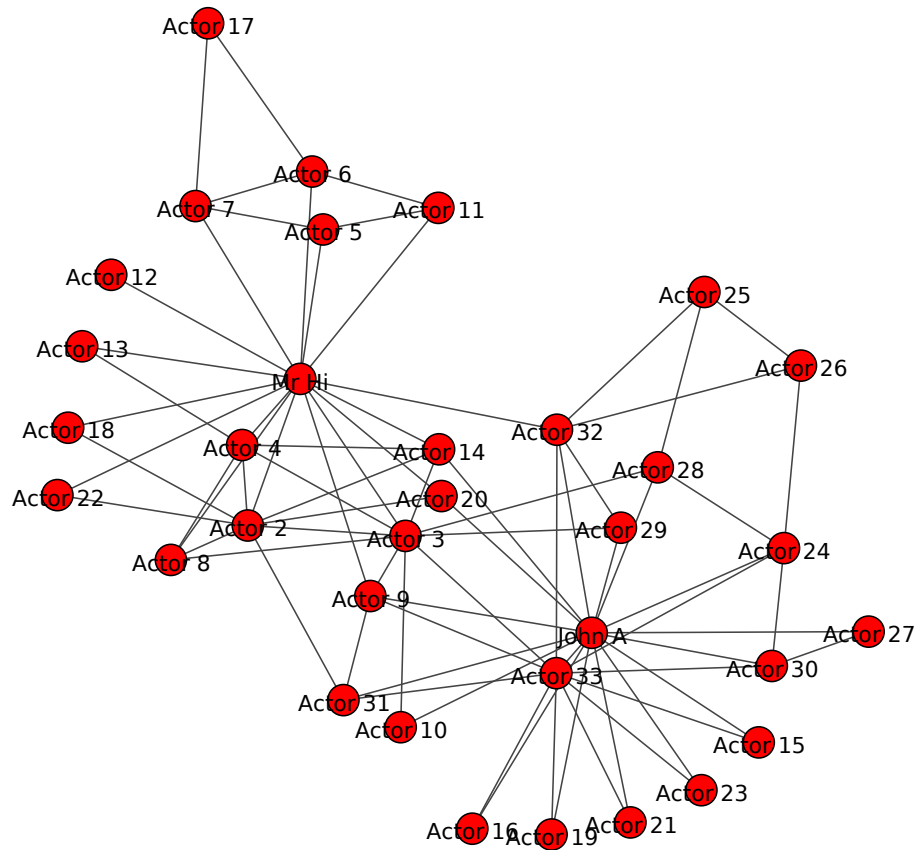
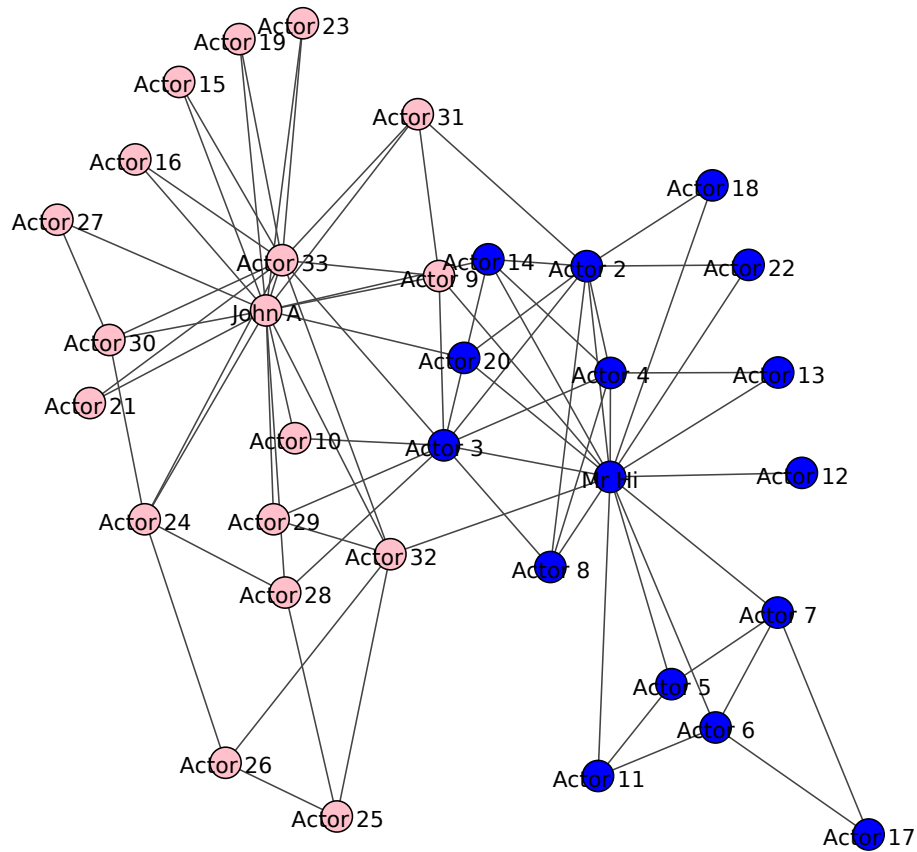
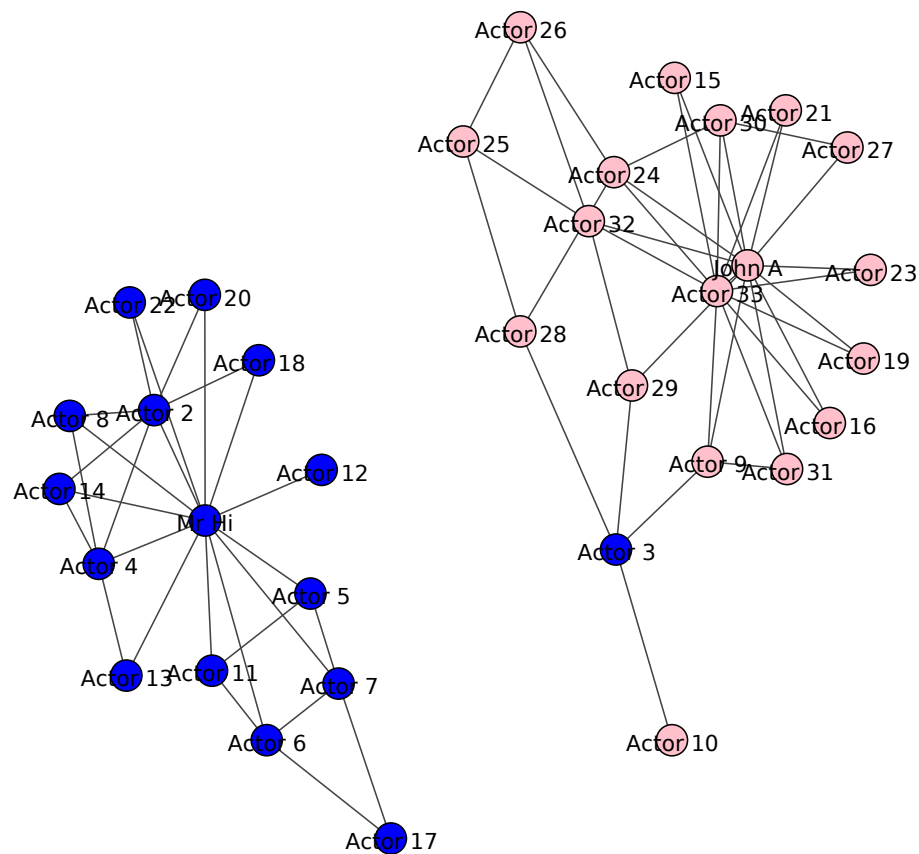


Figure 2: Original plot before split with colours showing actual final factions after split



According to the result shown in colours, the final result should be a total separation of the two distinct colours forming two clusters after the split. Colour represents reality ie how the ended up splitting up. Mr Hi and members are colored blue, while John A and his members are coloured pink.

Figure 3: Plot after split showing predicted two clusters



From the result it can be seen that the prediction by the algorithm was highly accurate with the exception of one node, (Actor 3). Hence the algorithm supports reality with a high degree of accuracy, 97 percent.

## Problem 2

2. We know the group split in two different groups. Suppose the disagreements in the group were more nuanced – what would the clubs look like if they split into groups of 3, 4, and 5?

Figure 3: Plot after split showing predicted three clusters

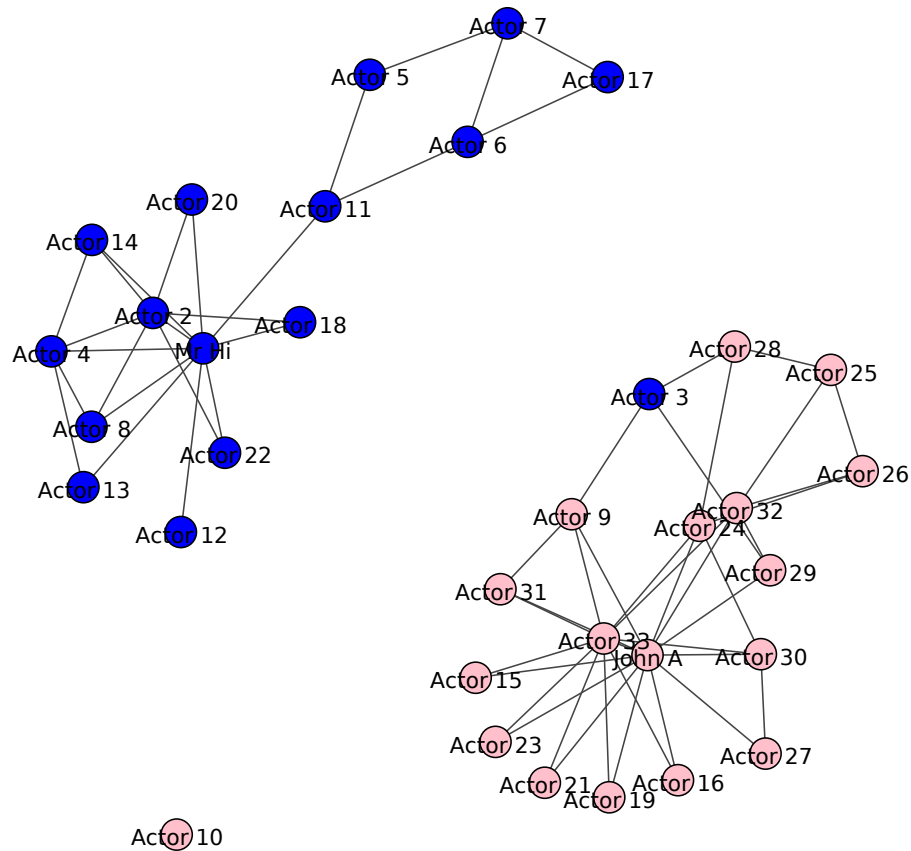




Figure 3: Plot after split showing predicted four clusters

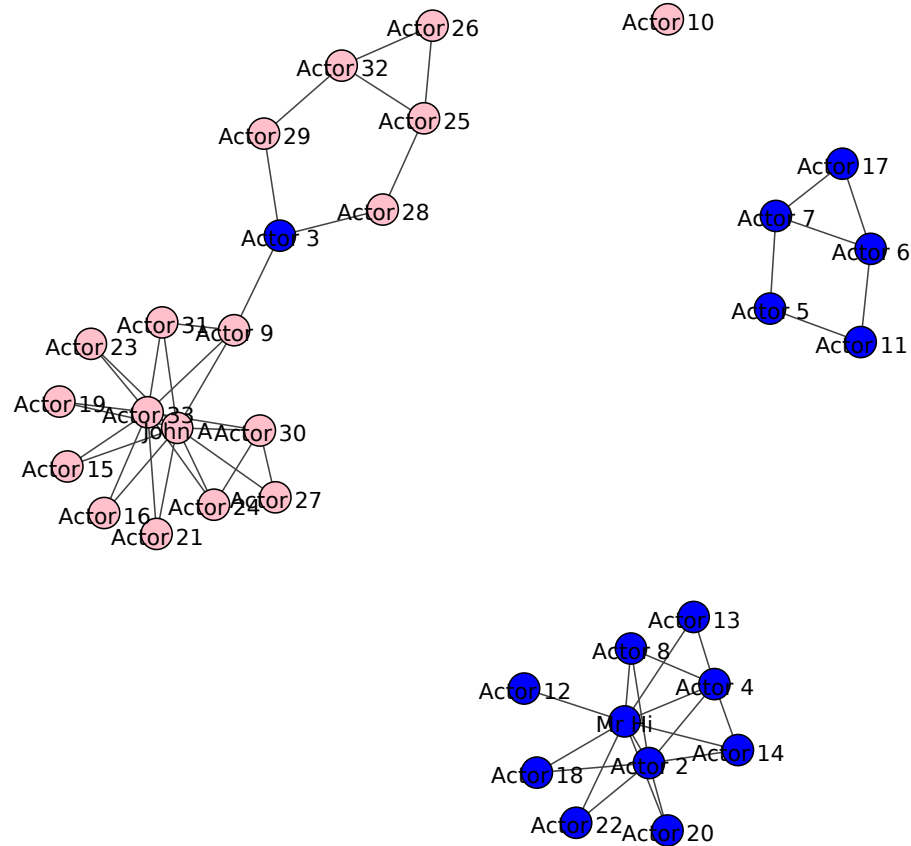
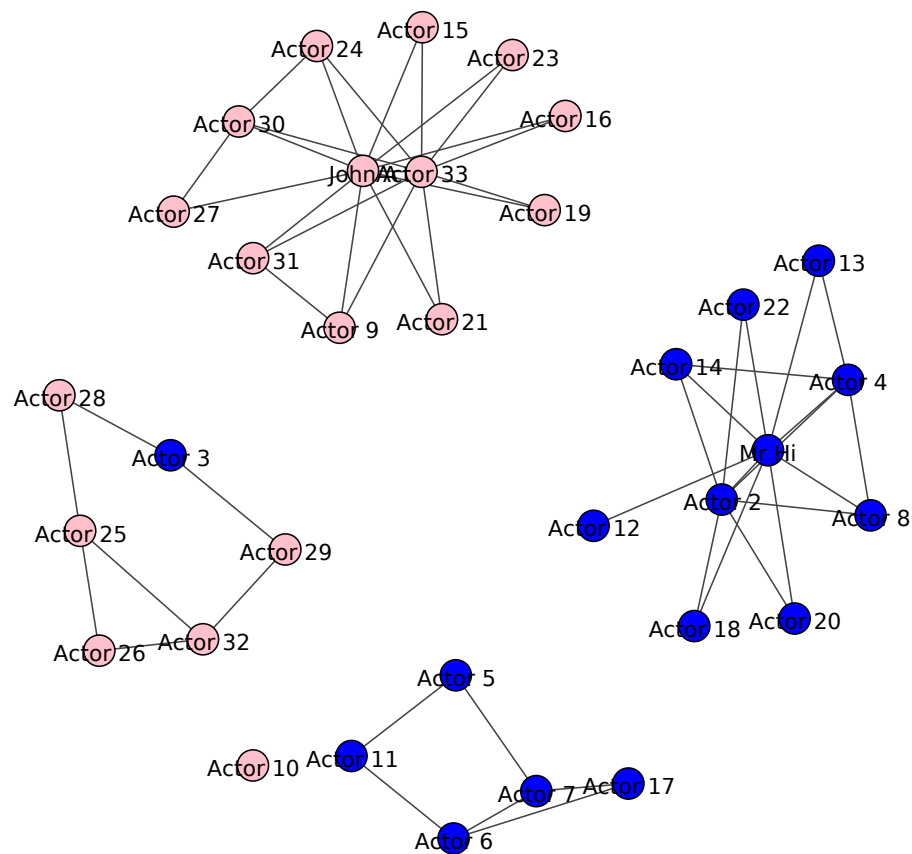


Figure 3: Plot after split showing predicted five clusters



Upon splitting into three clusters I noticed Actor10 was isolated. The colour coding I used also helped me see how the original 2 factions will be predicted to split further if they did.